XML-Based Data Preparation for Robust Deep Parsing

XML-Based Data Preparation for Robust Deep Parsing

Claire Grover and **Alex Lascarides**
Division of Informatics
The University of Edinburgh
2 Buccleuch Place
Edinburgh EH8 9LW, UK
{C.Grover, A.Lascarides}@ed.ac.uk

Abstract

We describe the use of XML tokenisation, tagging and mark-up tools to prepare a corpus for parsing. Our techniques are generally applicable but here we focus on parsing Medline abstracts with the ANLT wide-coverage grammar. Hand-crafted grammars inevitably lack coverage but many coverage failures are due to inadequacies of their lexicons. We describe a method of gaining a degree of robustness by interfacing POS tag information with the existing lexicon. We also show that XML tools provide a sophisticated approach to pre-processing, helping to ameliorate the 'messiness' in real language data and improve parse performance.

1 Introduction

The field of parsing technology currently has two distinct strands of research with few points of contact between them. On the one hand, there is thriving research on shallow parsing, chunking and induction of statistical syntactic analysers from treebanks; and on the other hand, there are systems which use hand-crafted grammars which provide both syntactic and semantic coverage. 'Shallow' approaches have good coverage on corpus data, but extensions to semantic analysis are still in a relative infancy. The 'deep' strand of research has two main problems: inadequate coverage, and a lack of reliable techniques to select the correct parse. In this paper we describe ongoing research which uses hybrid technologies to address the problem of inadequate coverage of a 'deep' parsing system. In Section 2 we describe how we have modified an existing hand-crafted grammar's look-up procedure to utilise part-of-speech (POS) tag information, thereby ameliorating the lexical information shortfall. In Section 3 we describe how we combine a variety of existing NLP tools to pre-process real data up to the point where a hand-crafted grammar can start to be useful. The work described in both sections is enabled by the use of an XML processing paradigm whereby the corpus is converted to XML with analysis results encoded as XML annotations. In Section 4 we report on an experiment with a random sample of 200 sentences which gives an approximate measure of the increase in performance we have gained.

The work we describe here is part of a project which aims to combine statistical and symbolic processing techniques to compute lexical semantic relationships, e.g. the semantic relations between nouns in complex nominals. We have chosen the medical domain because the field of medical informatics provides a relative abundance of pre-existing knowledge bases and ontologies. Our efforts so far have focused on the OHSUMED corpus (Hersh et al., 1994) which is a collection of Medline abstracts of medical journal papers.[1]

While the focus of the project is on semantic issues, a prerequisite is a large, reliably annotated corpus and a level of syntactic process-

[1] Sager et al. (1994) describe the Linguistic String Project's approach to parsing medical texts.

ing that supports the computation of semantics. The computation of 'grammatical relations' from shallow parsers or chunkers is still at an early stage (Buchholz et al., 1999, Carroll et al., 1998) and there are few other robust semantic processors, and none in the medical domain. We have therefore chosen to re-use an existing handcrafted grammar which produces compositionally derived underspecified logical forms, namely the wide-coverage grammar, morphological analyser and lexicon provided by the Alvey Natural Language Tools (ANLT) system (Carroll et al. 1991, Grover et al. 1993). Our immediate aim is to increase coverage up to a reasonable level and thereafter to experiment with ranking the parses, e.g. using Briscoe and Carroll's (1993) probabilistic extension of the ANLT software.

We use XML as the preprocessing mark-up technology, specifically the LT TTT and LT XML tools (Grover et al., 2000; Thompson et al., 1997). In the initial stages of the project we converted the OHSUMED corpus into XML annotated format with mark-up that encodes word tokens, POS tags, lemmatisation information etc. The research reported here builds on that mark-up in a further stage of pre-processing prior to parsing. The XML paradigm has proved invaluable throughout.

2 Improving the Lexical Component

2.1 Strategy

The ANLT grammar is a unification grammar based on the GPSG formalism (Gazdar et al., 1985), which is a precursor of more recent 'lexicalist' grammar formalisms such as HPSG (Pollard and Sag, 1994). In these frameworks lexical entries carry a significant amount of information including subcategorisation information. Thus the practical parse success of a grammar is significantly dependent on the quality of the lexicon. The ANLT grammar is distributed with a large lexicon which was derived semi-automatically from a machine-readable dictionary (Carroll and Grover, 1988). This lexicon is of varying quality: function words such as complementizers, prepositions, determiners and quantifiers are all reliably hand-coded but content words are less reliable. Verbs are generally coded to a high standard but the noun and adjective lexicons are full of redundancies and duplications. Since these duplications can lead to huge increases in the number of spurious parses, an obvious first step was to remove all duplications from the existing lexicons and to collapse certain ambiguities such as the count/mass distinction into single underspecified entries. A second critical step was to increase the character set that the spelling rules in the morphological analyser handle, so as to accept capitalised and non-alphabetic characters in the input.

Once these ANLT-internal problems are overcome, the main problem of inadequate lexical coverage still remains: if we try to parse OHSUMED sentences using the ANLT lexicon and no other resources, we achieve very poor results because most of the medical domain words are simply not in the lexicon and there is no 'robustness' strategy built into ANLT. One solution to this problem would be to find domain specific lexical resources from elsewhere and to merge the new resources with the existing lexicon. However, the resulting merged lexicon may still not have sufficient coverage and a means of achieving robustness in the face of unknown words would still be required. Furthermore, every move to a new domain would depend on domain-specific lexical resources being available. Because of these disadvantages, we have pursued an alternative solution which allows parsing to proceed without the need for extra lexical resources and with robustness built into the strategy. This alternative strategy does not preclude the use of domain specific lexical resources but it does provide a basic level of performance which further resources can be used to improve upon.

The strategy we have adopted relies first on sophisticated XML-based tokenisation (see Section 3) and second on the combination of POS tag information with the existing ANLT lexical resources. Our view is that POS tag information for content words (nouns, verbs, adjectives, adverbs) is usually reliable and informative, while tagging of function words (complementizers, determiners, particles, conjunctions, auxiliaries, pronouns, etc.) can be erratic and provides less information than the hand-written entries for function words that are typically developed side-by-side with wide coverage grammars. Furthermore, unknown words are far more likely to be con-

tent words than function words, so knowledge of the POS tag will most often be needed for content words. Our idea, then, is to tag the input but to retain only the content word POS tags and use them during lexical look-up in one of two ways. If the word exists in the lexicon then the POS tag is used to access only those entries of the same basic category. If, on the other hand, the word is not in the lexicon then a basic underspecified entry for the POS tag is used as the lexical entry for the word. In the first case, the POS tag is used as a filter, accessing only entries of the appropriate category and cutting down on the parser's search space. In the second case, the basic category of the unknown word is supplied and this enables parsing to proceed. For example, if the following partially tagged sentence is input to the parser, it is successfully parsed.[2]

```
We have developed_VBN a variable_JJ
suction_NN system_NN for irrigation_NN ,
aspiration_NN and vitrectomy_NN
```

Without the tags there would be no parse since the words *irrigation* and *vitrectomy* are not in the ANLT lexicon. Furthermore, tagging *variable* as an adjective ensures that the noun entry for *variable* is not accessed, thus cutting down on parse numbers (3 versus 6 in this case).

The two cases interact where a lexical entry is present in the ANLT lexicon but not with the relevant category. For example, *monitoring* is present in the ANLT lexicon as a verb but not as a noun:

```
We studied_VBD the value_NN of
transcutaneous_JJ carbon_NN dioxide_NN
monitoring_NN during transport_NN
```

Look up of the word_tag pair `monitoring_NN` fails and the basic entry for the tag `NN` is used instead. Without the tag, the verb entry for *monitoring* would be accessed and the parse would fail.

In the following example the adjectives *diminished* and *stabilized* exist only as verb entries: with the `JJ` tag the parse succeeds but without it, the verb entries are accessed and the parse fails.

```
There was radiographic_JJ evidence_NN of
diminished_JJ or stabilized_JJ pleural_JJ
effusion_NN
```

[2]The LT TTT tagger uses the Penn Treebank tagset (Marcus et al., 1994): JJ labels adjectives, NN labels nouns and VB labels verbs.

Note that cases such as these would be problematic for a strategy where tagging was used only when lexical look-up failed, since here lexical look-up doesn't fail, it just provides an incomplete set of entries. It is of course possible to augment the grammar and/or lexicon with rules to infer noun entries from *verb+ing* entries and adjective entries from *verb+ed* entries. However, this will increase lexical ambiguity quite considerably and lead to higher numbers of spurious parses.

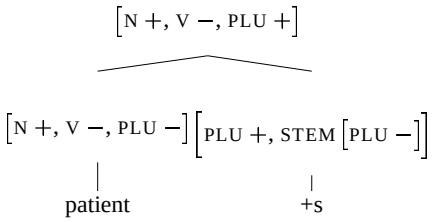
2.2 Implementation

We expect the technique outlined above to be applicable across a range of parsing systems. In this section we describe how we have implemented it within ANLT.

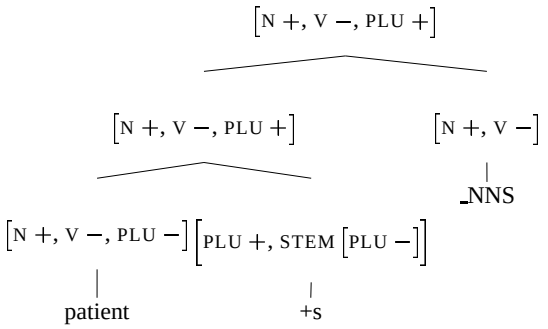
The version of the ANLT system described in Carroll et al. (1991) and Grover et al. (1993) does not allow tagged input but work by Briscoe and Carroll (1993) on statistical parsing uses an adapted version of the system which is able to process tagged input, ignoring the words in order to parse sequences of tags. We use this version of the system, running in a mode where 'words' are looked up according to three distinct cases:

- **word look-up**: the word has no tag and must be looked up in the lexicon (and if look-up fails, the parse fails)
- **tag look-up**: the word has a tag, look-up of the word_tag pair fails, but the tag has a special hand-written entry which is used instead
- **word_tag look-up**: the word has a tag and look-up of the word_tag pair succeeds.

The resources provided by the system already adequately deal with the first two cases but the third case had to be implemented. The existing morphological analysis software was relatively easily adapted to give the performance we required. The ANLT morphological analyser performs regular inflectional morphology using a unification grammar for combining morphemes and rules governing spelling changes when morphemes are concatenated. Thus a plural noun such as *patients* is composed of the morphemes *patient* and *+s* with the features on the top node being inherited partially from the noun and partially from the inflectional affix:



In dealing with word_tag pairs, we have used the word grammar to treat the tag as a novel kind of affix which constrains the category of the lexical entry it attaches to. We have defined morpheme entries for content word tags so they can be used by special word grammar rules and attached to words of the appropriate category. Thus `patient_NN` is analysed using the noun entry for *patient* but not the adjective entry. Tag morphemes can be attached to inflected as well as to base forms, so the string `patients_NNS` has the following internal structure:



In defining the rules for word_tag pairs, we were careful to ensure that the resulting category would have exactly the same feature specification as the word itself. Thus the tag morpheme is specified only for basic category features which the word grammar requires to be shared by word and tag. All other feature specifications on the covering node are inherited from the word, not the tag. This method of combining POS tag information with lexical entries preserves all information in the lexical entries, including inflectional and subcategorisation information. The preservation of subcategorisation information is particularly necessary since the ANLT lexicon makes sophisticated distinctions between different subcategorisation frames which are critical for obtaining the correct parse and associated logical form.

3 XML Tools for Pre-Processing

The techniques described in this section, and those in the previous section, are made possible by our use of an XML processing paradigm throughout. We use the LT TTT and LT XML tools in pipelines where they add, modify or remove pieces of XML mark-up. Different combinations of the tools can be used for different processing tasks. Some of the XML programs are rule-based while others use maximum entropy modelling.

We have developed a pipeline which converts OHSUMED data into XML format and adds linguistic annotations. The early stages of the pipeline segment character strings first into words and then into sentences while subsequent stages perform POS tagging and lemmatisation. A sample part of the output of this basic pipeline is shown in Figure 1. The initial conversion to XML and the identification of words is achieved using the core LT TTT program *fsgmatch*, a general purpose transducer which processes an input stream and rewrites it using rules provided in a grammar file. The identification of sentence boundaries, mark-up of sentence elements and POS tagging is done by the statistical program *ltpos* (Mikheev, 1997). Words are marked up as `W` elements with further information encoded as values of attributes on the `W` elements. In the example, the `P` attribute's value is a POS tag and the `LM` attribute's is a lemma (only on nouns and verbs). The lemmatisation is performed by Minnen et al.'s (2000) *morpha* program which is not an XML processor. In such cases we pass data out of the pipeline in the format required by the tool and merge its output back into the XML mark-up. Typically we use McKelvie's (1999) *xmlperl* program to convert out of and back into XML: for ANLT this involves putting each sentence on one line, converting some `W` elements into word_tag pairs and stripping out all other XML mark-up to provide input to the parser in the form it requires. We are currently experimenting with bringing the labelled bracketing of the parse result back into the XML as 'stand-off' mark up.

3.1 Pre-Processing for Parsing

In Section 2 we showed how POS tag mark-up could be used to add to existing lexical resources. In this section we demonstrate how the

<RECORD>
<ID>395</ID>
<MEDLINE-ID>87052477</MEDLINE-ID>
<SOURCE>Clin Pediatr (Phila) 8703; 25(12):617-9 </SOURCE>
<MESH>
Adolescence; Alcoholic Intoxication/BL/*EP; Blood Glucose/AN; Canada; Child; Child, Preschool; Electrolytes/BL; Female; Human; Hypoglycemia/ET; Infant; Male; Retrospective Studies.
</MESH>
<TITLE>Ethyl alcohol ingestion in children. A 15-year review.</TITLE>
<PTYPE>JOURNAL ARTICLE.</PTYPE>
<ABSTRACT>
<SENTENCE><W P='DT'>A</W> <W P='JJ'>retrospective</W>
<W P='NN' LM='study'>study</W> <W P='VBD' LM='be'>was</W>
<W P='VBN' LM='conduct'>conducted</W> <W P='IN'>by</W> <W P='NN' LM='chart'>chart</W>
<W P='NNS' LM='review'>reviews</W> <W P='IN' >of</W> <W P='CD'>27</W>
<W P='NNS' LM='patient'>patients</W> <W P='IN'>with</W> <W P='JJ'>documented</W> <W P='NN' LM='ethanol'>ethanol</W> <W P='NN' LM='ingestion'>ingestion</W><W P='.'>.</W>
</SENTENCE> <SENTENCE> . . . </SENTENCE> <SENTENCE> . . . </SENTENCE>
</ABSTRACT>
<AUTHOR>Leung AK.</AUTHOR>
</RECORD>

Figure 1: A sample from the XML-marked-up OHSUMED corpus

XML approach allows for flexibility in the way data is converted from marked-up corpus material to parser input. This method enables 'messy' linguistic data to be rendered innocuous prior to parsing, thereby avoiding the need to make hand-written low-level additions to the grammar itself.

3.1.1 Changing POS tag labels

One of the failings of the ANLT lexicon is in the subcategorisation of nouns: each noun has a zero subcategorisation entry but many nouns which optionally subcategorise a complement lack the appropriate entry. For example, the nouns *use* and *management* do not have entries with an *of*-PP subcategorisation frame so that in contexts where an *of*-PP is present, the correct parse will not be found. The case of *of*-PPs is a special one since we can assume that whenever *of* follows a noun it marks that noun's complement. We can encode this assumption in the layer of processing that converts the XML mark-up to the format required by the parser: an *fsgmatch* rule changes the value of the P attribute of a noun from NN to NNOF or from NNS to NNSOF whenever it is followed by *of*. By not adding morpheme entries for NNOF and NNSOF we ensure that word_tag look-up will fail and the system will fall back on tag look-up using special entries for NNOF and NNSOF which have only an *of*-PP subcategorisation frame. In this way the parser will be forced to attach *of*-PPs following nouns as their complements.

3.1.2 Numbers, formulae, etc.

Although we have stated that we only retain content word tags, in practice we also retain certain other tags for which we provide no morpheme entry in the morphological system so as to achieve tag rather than word_tag look-up. For example, we retain the CD tag assigned to numerals and provide a general purpose entry for it so that sentences containing numerals can be parsed without needing lexical entries for them. We also use a pre-existing tokenisation component which recognises spelled out numbers to which the CD tag is also assigned:

<W P='CD'>thirty-five</W>	thirty-five_CD
<W P='CD'>Twenty one</W>	Twenty∼one_CD
<W P='CD'>176</W>	176_CD

The program *fsgmatch* can be used to group words together into larger units using handwritten rules and small lexicons of 'multi-word' words. For the purposes of parsing, these larger units can be treated as words, so the grammar does not need to contain special rules for 'multi-word' words:

<W P='IN'>In order to</W>	In∼order∼to_IN
<W P='IN'>in relation to</W>	in∼relation∼to_IN
<W P='JJ'>in vitro</W>	in∼vitro_JJ

The same technique can be used to package up a wide variety of formulaic expressions which would cause severe problems to most hand-crafted grammars. Thus all of the following 'words' have been identified using *fsgmatch* rules and can be passed to the parser as unanalysable chunks.[3] The classification of the examples below as nouns reflects a working hypothesis that they can slot into the correct parse as noun phrases but there is room for experimentation since the conversion to parser input format can rewrite the tag in any way. It may turn out that they should be given a more general tag which corresponds to several major category types.

<W P='NN'>P less than 0.001</W>
<W P='NN'>166 +/- 77 mg/dl</W>
<W P='NN'>2 to 5 cc/day</W>
<W P='NN'>9.1 v. 5.1 ml</W>
<W P='NN'>2.5 mg i.v.</W>

It is important to note that our method of dividing the labour between pre-processing and parsing allows for experimentation to get the best possible balance. We are still developing our formula recognition subcomponent which has so far been entirely hand-coded using *fsgmatch* rules. We believe that it is more appropriate to do this hand-coding at the pre-processing stage rather than with the relatively unwieldy formalism of the ANLT grammar. Moreover, use of the XML paradigm might allow us to build a component that can induce rules for regular formulaic expressions thus reducing the need for hand-coding.

3.1.3 Dealing with tagger errors

The tagger we use, *ltpos*, has a reported performance comparable to other state-of-the-art taggers. However, all taggers make errors, especially when used on data different from their training data. With the strategy outlined in this paper, where we only retain a subset of tags, many tagging errors will be harmless. However, content word tagging errors will be detrimental since the basic noun/verb/adjective/adverb distinction drives lexical look-up and only entries of the same category as the tag will be accessed. If we find that the tagger consistently makes the same error in a particular context, for example mistagging *+ing* nominalisations as verbs (VBG), then we can use *fsgmatch* rules to replace the tag in just those contexts. The new tag can be given a definition which is ambiguous between NN and VBG, thereby ensuring that a parse can be achieved.

A second strategy that we are exploring involves using more than one tagger. Our current pipeline includes a call to Elworthy's (1994) CLAWS2 tagger. We encode the tags from this tagger as values of the attribute C2 on words:

<W P='NNS' C2='NN2' LM='case'>cases</W>
<W P='VBN' C2='VVN' LM='find'>found</W>

Many mistaggings can be found by searching for words where the two taggers disagree and they can be corrected in the mapping from XML format to parser input by assigning a new tag which is ambiguous between the two possibilities. For example, *ltpos* incorrectly tags the word *bound* in the following example as a noun but the CLAWS2 tagger correctly categorises it as a verb.

```
a large_JJ body_NNOF of hemoglobin_NN
bound_NNVVN to the ghost_NN membrane_NN
```

We use *xmlperl* rules to map from XML to ANLT input and reassign these cases to the 'composite' tag NNVVN, which is given both a noun and a verb entry. This allows the correct parse to be found whichever tagger is correct. An alternative approach to the mistagging problem would be to use just one tagger which returns multiple tags and to use the relative probability of the tags to determine cases where a composite tag could be created in the mapping to parser input. Charniak et al. (forthcoming) reject a multiple tag approach when using a probabilistic context-free-grammar parser, but it is unclear whether their result is relevant to a hand-crafted grammar.

3.2 An XML corpus

There are numerous advantages to working with XML tools. One general advantage is that we can add linguistic annotations in an entirely automatic and incremental fashion, so as to produce a heavily annotated corpus which may well prove useful to a number of researchers for a number of linguistic activities. In the work described here we have not used any domain specific information. However, it would clearly be possible to add domain specific information as further annotations

[3] Futrelle et al. (1991) discuss tokenisation issues in biological texts.

using such resources as UMLS (UMLS, 2000). Indeed, we have begun to utilise UMLS and hope to improve the accuracy of the existing mark-up by incorporating lexical and semantic information. Since the annotations we describe are computed entirely automatically, it would be a simple matter to use our system to mark up new Medline data to increase the size of our corpus considerably.

A heavily annoted corpus quickly becomes unreadable but if it is an XML annotated corpus then there are several tools to help visualise the data. For example, we use *xmlperl* to convert from XML to HTML to view the corpus in a browser.

4 Evaluation and Future Research

With a corpus such as OHSUMED where there is no gold-standard tagged or hand-parsed subpart, it is hard to reliably evaluate our system. However, we did an experiment on 200 sentences taken at random from the corpus (average sentence length: 21 words). We ran three versions of our pre-processor over the 200 sentences to produce three different input files for the parser and for each input we counted the sentences which were assigned at least one parse. All three versions started from the same basic XML annotated data, where words were tagged by both taggers and parenthesised material was removed. Version 1 converted from this format to ANLT input simply by discarding the mark-up and separating off punctuation. Version 2 was the same except that content word POS tags were retained. Version 3 was put through our full pipeline which recognises formulae, numbers etc. and which corrects some tagging errors. The following table shows numbers of sentences successfully parsed with each of the three different inputs:

	Version 1	**Version 2**	**Version 3**
Parses	4 (2%)	32 (16%)	79 (39.5%)

The extremely low success rate of Version 1 is a reflection of the fact that the ANLT lexicon does not contain any specialist lexical items. In fact, of the 200 sentences, 188 contained words that were not in the lexicon, and of the 12 that remained, 4 were successfully parsed. The figure for Version 2 gives a crude measure of the contribution of our use of tags in lexical look-up and the figure for Version 3 shows further gains when further pre-processing techniques are used.

Although we have achieved an encouraging overall improvement in performance, the total of 39.5% for Version 3 is not a precise reflection of accuracy of the parser. In order to determine accuracy, we hand-examined the parser output for the 79 sentences that were parsed and recorded whether or not the *correct* parse was among the parses found. Of these 79 sentences, 61 (77.2%) were parsed correctly while 18 (22.8%) were not, giving a total accuracy measure of 30.5% for Version 3. While this figure is rather low for a practical application, it is worth reiterating that this still means that nearly one in three sentences are not only correctly parsed but they are also assigned a logical form. We are confident that the further work outlined below will achieve an improvement in performance which will lead to a useful semantic analysis of a significant proportion of the corpus. Furthermore, in the case of the 18 sentences which were parsed incorrectly, it is important to note that the 'wrong' parses may sometimes be capable of yielding useful semantic information. For example, the grammar's compounding rules do not yet include the possibility of coordinations within compounds so that the NP *the MS and direct blood pressure methods* can only be wrongly parsed as a coordination of two NPs. However, the rest of the sentence in which the NP occurs is correctly parsed.

An analysis of the 18 sentences which were parsed incorrectly reveals that the reasons for failure are distributed evenly across three causes: a word was mistagged and not corrected during preprocessing (6); the segmentation into tokens was inadequate (5); and the grammar lacked coverage (7). A casual inspection of a random sample of 10 of the sentences which failed to parse at all reveals a similar pattern although for several there were multiple reasons for failure. Lack of grammatical coverage was more in evidence, perhaps not surprisingly since work on tuning the grammar to the domain has not yet been done.

Although we are only able to parse between 30 and 40 percent of the corpus, we will be able to improve on that figure quite considerably in the future through continued development of the pre-processing component. Moreover, we have not yet incorporated any domain specific lexical

knowledge from, e.g., UMLS but we would expect this to contribute to improved performance. Furthermore, our current level of success has been achieved without significant changes to the original grammar and, once we start to tailor the grammar to the domain, we will gain further significant increases in performance. As a final stage, we may find it useful to follow Kasper et al. (1999) and have a 'fallback' strategy for failed parses where the best partial analyses are assembled in a robust processing phase.

References

T. Briscoe and J. Carroll. 1993. Generalised probabilistic LR parsing of natural language (corpora) with unification grammars. *Computational Linguistics*, 19(1):25–60.

S. Buchholz, J. Veenstra, and W. Daelemans. 1999. Cascaded grammatical relation assignment. In *EMNLP '99*, pp 239–246, Maryland.

J. Carroll and C. Grover. 1988. The derivation of a large computational lexicon of English from LDOCE. In B. Boguraev and E. J. Briscoe, editors, *Computational Lexicography for Natural Language Processing*. Longman, London.

J. Carroll, T. Briscoe, and C. Grover. 1991. A development environment for large natural language grammars. Technical Report 233, Computer Laboratory, University of Cambridge.

J. Carroll, T. Briscoe, and G. Minnen. 1998. Can subcategorisation probabilities help a statistical parser? In *Proceedings of the 6th ACL/SIGDAT Workshop on Very Large Corpora*, pp 118–126, Montreal. ACL/SIGDAT.

E. Charniak, G. Carroll, J. Adcock, A. Cassandra, Y. Gotoh, J. Katz, M. Littman, and J. McCann. forthcoming. Taggers for parsers. *Artificial Intelligence*.

D. Elworthy. 1994. Does Baum-Welch re-estimation help taggers? In *Proceedings of the 4th ACL conference on Applied Natural Language Processing*, pp 53–58, Stuttgart, Germany.

R. Futrelle, C. Dunn, D. Ellis, and M. Pescitelli. 1991. Preprocessing and lexicon design for parsing technical text. In *2nd International Workshop on Parsing Technologies (IWPT-91)*, pp 31–40, Morristown, New Jersey.

G. Gazdar, E. Klein, G. Pullum, and I. Sag. 1985. *Generalized Phrase Structure Grammar*. Basil Blackwell, London.

C. Grover, J. Carroll, and T. Briscoe. 1993. The Alvey Natural Language Tools grammar (4th release). Technical Report 284, Computer Laboratory, University of Cambridge.

C. Grover, C. Matheson, A. Mikheev, and M. Moens. 2000. LT TTT—a flexible tokenisation tool. In *LREC 2000—Proceedings of the Second International Conference on Language Resources and Evaluation, Athens*, pp 1147–1154.

W. Hersh, C. Buckley, TJ Leone, and D. Hickam. 1994. OHSUMED: an interactive retrieval evaluation and new large test collection for research. In W. Bruce Croft and C. J. van Rijsbergen, editors, *Proceedings of the 17th Annual International Conference on Research and Development in Information Retrieval*, pp 192–201, Dublin, Ireland.

W. Kasper, B. Kiefer, H.-U. Krieger, C.J. Rupp, and K. Worm. 1999. Charting the depths of robust speech parsing. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pp 405–412, Maryland.

M. Marcus, G. Kim, M. Marcinkiewicz, R. MacIntyre, A. Bies, M. Ferguson, K. Katz, and B. Schasberger. 1994. The Penn treebank: annotating predicate argument structure. In *ARPA Human Language Technologies Workshop*.

D. McKelvie. 1999. XMLPERL 1.0.4. XML processing software. `http://www.cogsci.ed.ac.uk/~dmck/xmlperl`.

A. Mikheev. 1997. Automatic rule induction for unknown word guessing. *Computational Linguistics*, 23(3):405–423.

G. Minnen, J. Carroll, and D. Pearce. 2000. Robust, applied morphological generation. In *Proceedings of 1st International Natural Language Conference (INLG '2000)*, Mitzpe Ramon, Israel.

C. Pollard and I. Sag. 1994. *Head-Driven Phrase Structure Grammar*. CSLI and University of Chicago Press, Stanford, Ca. and Chicago, Ill.

N. Sager, M. Lyman, C. Bucknall, N. Nhan, and L. J. Tick. 1994. Natural language processing and the representation of clinical data. *Journal of the American Medical Informatics Association*, 1(2):142–160.

H. Thompson, R. Tobin, D. McKelvie, and C. Brew. 1997. LT XML. Software API and toolkit for XML processing. `http://www.ltg.ed.ac.uk/software/`.

UMLS. 2000. *Unified Medical Language System (UMLS) Knowledge Sources*. National Library of Medicine, Bethesda (MD), 11th edition.